

9 Un año más de hacer comunidad

Revista de Software Libre ATIX 2009

Colaboradores

Equipo Editorial

Dirección y Coordinación General

Esteban Saavedra López (esteban.saavedra@atixlibre.org)

Diseño y Maquetación

Jenny Saavedra López (jenny.saavedra@atixlibre.org)

Esteban Saavedra López (esteban.saavedra@atixlibre.org)

Revisiones

Esteban Saavedra López

Jenny Saavedra López

Noticias

Jenny Saavedra López

Marcia Velasquez

Autores Frecuentes

Esteban Saavedra López

Joseph Sandoval

Mario Carrion

Martín Márquez

Renzo Martinez

Rafael Rendon

Herramientas

La edición de esta revista fue realizada de forma integra haciendo uso de Software Libre









Palabra quechua,

con un sentimiento profundo

y con gran significado filosófico

El que lo sabe

El que lo intenta

El que lo puede

El que lo logra

Editorial

Transcurrieron 365 días más en nuestras vidas, 365 días más de estar inmersos en nuestras actividades, pues como cada año, es hora de hacer un recuento de todas las cosas, de todas la actividades, de todas las experiencias que tuvimos, para poder evaluarlas y ver cuanto crecimos o cuanto aprendimos. Para muchos es bastante triste ver que se va un año más, por que represento un año bueno; en cambio otros sentirán el alivio de ver que se va un año que fue malo para ellos, pero todos esperanzados esperamos que el año que viene sea mejor que el que se va.

Un año más de hacer comunidad, un título que refleja nuestra dedicación dentro la comunidad nacional e internacional, una comunidad que desde que emprendimos el desafío nos dio su respaldo y confió en nuestro trabajo. A toda esta gran comunidad les damos nuestro agradecimiento y les reiteramos nuestro compromiso de seguir trabajando promoviendo el uso de las tecnologías libres y destacando el concepto de libertad.

En este ultimo número del 2009, deseamos expresarles nuestros mayores deseos de felicidad en estas fiestas de fin de año, y la mayor prosperidad en el año que viene. “**FELICIDADES A TODA LA COMUNIDAD**”

Un año más de hacer comunidad.

Bienvenidos a nuestro décimo quinto número

Esteban Saavedra López
Director y Coordinador General

Contenido

Liberado el 28 de diciembre de 2009



Educación pública, software privativo



La constitución política Costarricense en su Artículo 78, capítulo VII denominado Educación y Cultura indica:

La educación preescolar y la general básica son obligatorias. Estas y la educación diversificada en el sistema público son gratuitas y costeadas por la Nación.

Es interesante pensar que la inclusión de nuestros jóvenes en un sistema educativo moderno en el que el uso de las tecnologías sea el medio para lograr un justo acceso al conocimiento, basados en el compromiso del estado plasmado en nuestra constitución de ofrecer educación gratuita y obligatoria denota graves incongruencias en sus efectos prácticos tal y como se realiza hoy en día.

Parece no tener sentido que en las escasas aproximaciones de nuestros niÒos a las computadoras se realicen sobre software privativo. El software privativo es aquel que su licencia no permite copiar, distribuir o modificar el programa como tal.

En la mayoría de los casos el software privativo no permite ser copiado o instalado en múltiples computadoras. Así por ejemplo si a un niÒo se le enseÒa en su centro educativo utilizar un paquete de ofimática privativo. øQuÈ posibilidades tiene ese estudiante de instalar dicho software en su casa? La respuesta tiene dos enfoques, el más difundido en nuestro país sería conseguir una copia ilegal del software, el segundo sería adquirir una licencia legal.

Si consideramos que el salario mínimo aprobado en el primer trimestre de 2009 para un mensajero es de ₡192.385 øSerá posible que se costee la compra de una única licencia por ₡60.000 de un paquete de ofimática reducido? øAcaso se cumplirá lo que se busca en nuestra constitución?.

El enfoque errado tomado por las autoridades partícipes de la gestión de la educación pública toma matices más preocupantes si tomamos el rumbo seguido. La táctica tomada es no tener rumbo; mientras algunos hablan de neutralidad tecnológica, es evidente que es descabellado no tener políticas claras de inclusión de la tecnología en la educación.

El impacto del software privativo en la educación pública no solo afecta económicamente. El software privativo no permite ser modificado por terceros, lo que provoca que nuestro modelo de inclusión de las tecnologías dependa de un único proveedor. A su vez la utilización de software privativo deja de lado las posibles mejoras que pueden ofrecer los centros de educación superior, ya que nunca podrían estudiar los programas y mucho menos publicar mejoras.

Hay que dejar claro que sí existen alternativas, existe software educativo de altísima calidad liberado bajo el esquema de software libre. El software libre es el que permite estudiar, modificar, distribuir el software basados en un enfoque colaborativo.

La utilización de software libre permitiría a los centros educativos regalar una copia del software a sus estudiantes sin necesidad de se incurra en un gasto adicional, permite a distintos entes contribuir en el desarrollo de

las aplicaciones y promueve la colaboración.

Lo que tal vez en otras organizaciones sea más complejo, en el caso de la educación pública la utilización de software libre es una necesidad real. Asociado a esto lo más importe es la definición políticas de gestión de utilización de software en nuestros centros educativos que permitan la inclusión a la tecnología a nuestros niÒos.

La táctica tomada es no tener rumbo; mientras algunos hablan de neutralidad tecnológica, es evidente que es descabellado no tener políticas claras de inclusión de la tecnología en la educación.

Autor

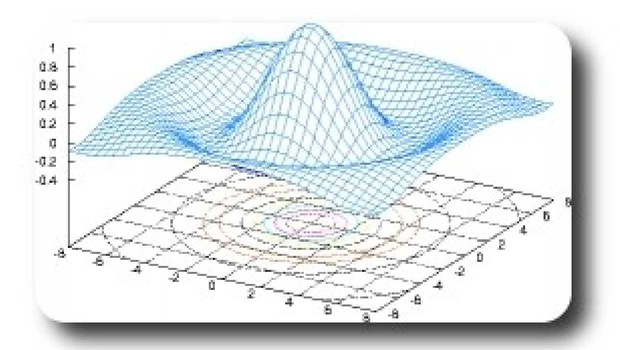


Allam Chaves Zamora

Entusiasta de Software Libre Costa Rica

GNUPlot una alternativa libre para graficar funciones

GNUPlot es un programa muy flexible para generar gráficas de funciones y datos. Su compatibilidad con casi todos los sistemas operativos, ha permitido que su uso se extienda, considerando además su sencillez, rápido aprendizaje y robustez.



Introducción

Son muchas las áreas de aplicación donde se precisan contar con representaciones gráficas de los datos, dentro del mundo científico y acadÈmico si bien existe un amplia variedad de programas orientados a este aspecto, podemos asegurar que **GNUPlot** se ha convertido en la solución por defecto al momento de requerir realizar gráficas de datos.

GNUPlot al ser una aplicación dentro el mundo del software libre, su uso no necesariamente se ha enfrascado en entornos como **GNULinux**, sino más al contrario se ha venido utilizando en casi todos los sistemas operativos.

Características

- ✔ Produce resultados tanto en pantalla, como en diversos formatos (PNG, EPS, SVG, JPEG, etc).
- ✔ Posibilidad de manejo de forma interactiva o por medio de scripts
- ✔ Permite interactuar con muchísimos programas matemáticos, estadísticos, económicos, etc

Requisitos

Actualmente todas las distribuciones de Linux, lo incluyen por defecto, en caso de no estar presente podemos hacer uso del gestor de paquetes propio de cada distribución:

- ✔ YUM para CentOS, Fedora
- ✔ apt-get para Debian y derivados

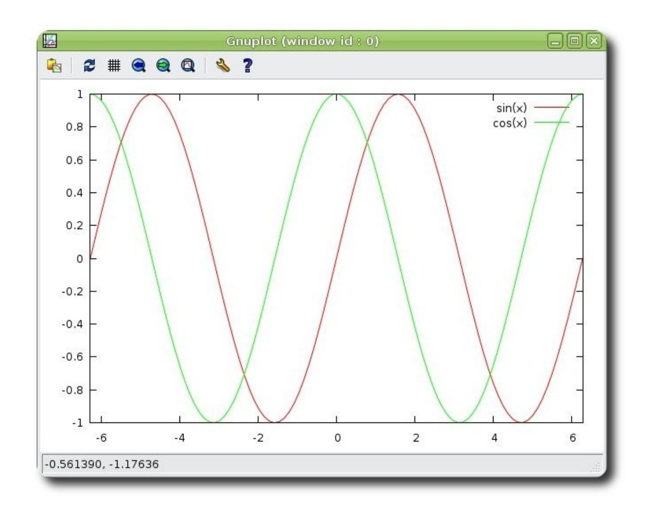
Funciones

Función	Descripción
abs(x)	valor absoluto de x, |x|
acos(x)	arco coseno de x
asin(x)	arco seno de x
atan(x)	arco tangente de x
cos(x)	coseno de x, x expresado en radianes.
cosh(x)	coseno hiperbólico de x, x expresado en radianes
erf(x)	función de error de x
exp(x)	función exponencial de x, base e
inverf(x)	función de error inverso de x

invnorm(x)	distribución normal inversa de x
log(x)	logaritmo de x, base e
log10(x)	logaritmo de x, base 10
norm(x)	función de distribución de Gauss normal
rand(x)	generador de números pseudo aleatorios
sgn(x)	1 If x > 0, -1 if x < 0, 0 if x=0
sin(x)	seno de x, x expresado en radianes
sinh(x)	seno hiperbólico de x, x expresado en radianes
sqrt(x)	raíz cuadrada de x
tan(x)	tangente de x, x expresado en radianes
tanh(x)	tangente hiperbólica de x, x expresado en radianes

Gráficas de funciones

```
plot [-2*pi:2*pi] sin(x),cos(x)
```



Uso de constantes

GNUPlot al igual que la mayoría de los lenguajes tiene la posibilidad de hacer uso de constantes, algunos ejemplos sencillos son mostrados a continuación:

```
a = 10
b = 5
y = a*x+b
```

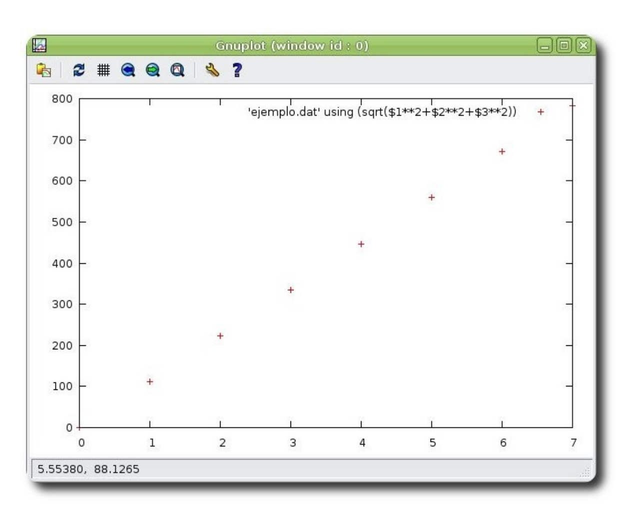
Gráficas de datos externos

En la mayoría de los casos, los usuarios tenemos la necesidad de hacer uso de datos que están almacenados de forma externa, **GNUPlot** permite acceder a estos datos siempre y cuando cumplan ciertas características básicas como ser en un archivo texto separado por tabuladores, dentro el cual pueden existir una serie de columnas que puede ser utilizadas de forma indistinta para realizar las gráficas que sean necesarias. Al momento de hacer uso de estos datos tenemos la posibilidad de seleccionar un conjunto de columnas y/o realizar operaciones entre ellas para poder ser utilizadas como entradas para los gráficas correspondientes.

En el ejemplo siguiente se muestra un ejemplo de un archivo de datos (**`ejemplo.dat`**).

```
0.000      0     0
0.001    100    50
0.002    200   100
0.003    300   150
0.004    400   200
0.005    500   250
0.006    600   300
0.007    700   350
```

Para utilizar alguna columna específica desde un archivo de datos podemos hacer uso de: **`plot ’ejemplo.dat’ using 1:2`** de forma directa, y en caso de requerir utilizar alguna columna dentro de un cálculo podríamos hacer uso de: **`plot ’ejemplo.dat’ using (sqrt($1**2+$2**2+$3**2))`**



GNUPlot Script

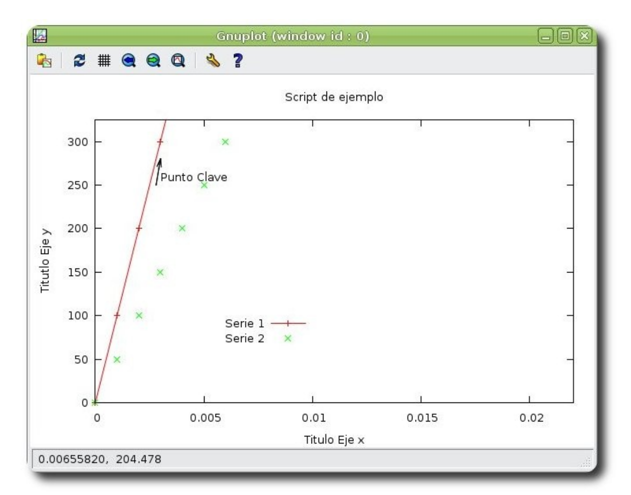
Muchas veces el realizar una gráfica requiere realizar un conjunto de pasos previos, reflejados en un conjunto de instrucciones, para este caso **GNUPlot** permite que ese conjunto de instrucciones sea almacenado en un archivo de texto (script) y sea invocado de forma interactiva o como parámetro de la instrucción de ploteo.

```
load 'archivo_script' (de forma
interactiva)
```

o

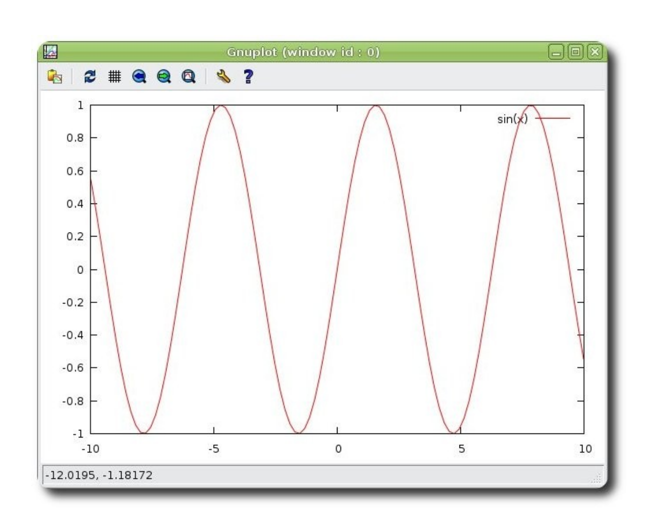
```
gnuplot “archivo_script” (como argumento)
```

```
#Gnuplot script de ejemplo
set autoscale
unset label  #remove any log-scaling
set xtic auto
set ytic auto
set title "Script de ejemplo"
set xlabel "Titulo Eje x"
set ylabel "Titutlo Eje y"
set key 0.01,100
set label "Punto Clave" at 0.003,260
set arrow from 0.0028, 250 to 0.003,280
set xr [0.0:0.022]
set yr [0:325]
plot "ejemplo.dat" using 1:2 t 'Serie 1'
with linespoints,"ejemplo.dat" using 1:3
t 'Serie 2' with points
pause 5 "Wait for 5 seconds"
```

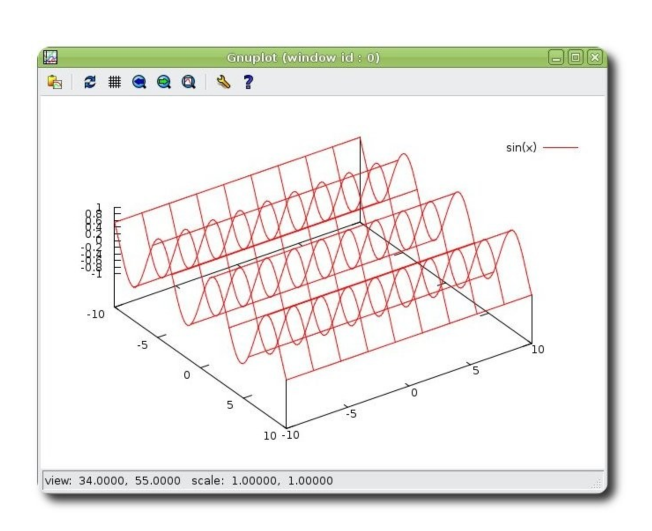


Gráficas 2D y 3D

```
plot sin(x)
```




```
splot sin(x)
```



Estilo de gráficas

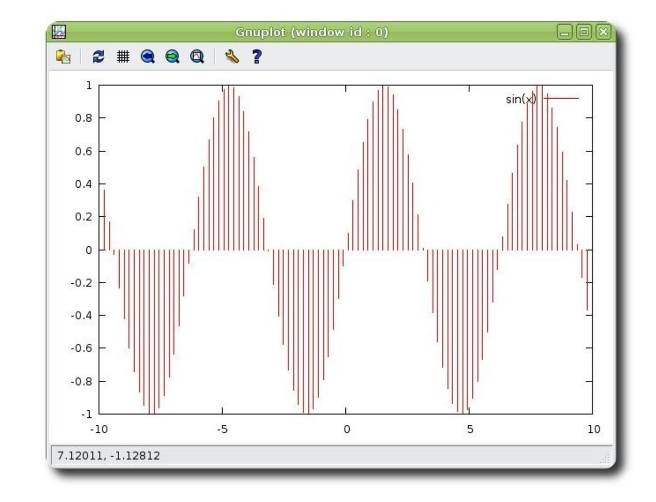
GNUPlot permite la personalización de estilos, más propiamente la personalización del estilo de trazo que se utiliza para las gráficas.

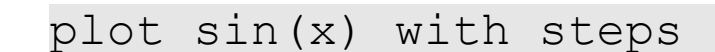
Los estilos disponibles son:

lines	points
impulses	yerrorbars
fsteps	histeps
boxxyerrorbars	vector
error lines	xerrorlines
linespoints	dots
xyerrorbars	steps
boxes	boxerrorbars
financebars	candlesticks
yerrorlines	xyerrorlines

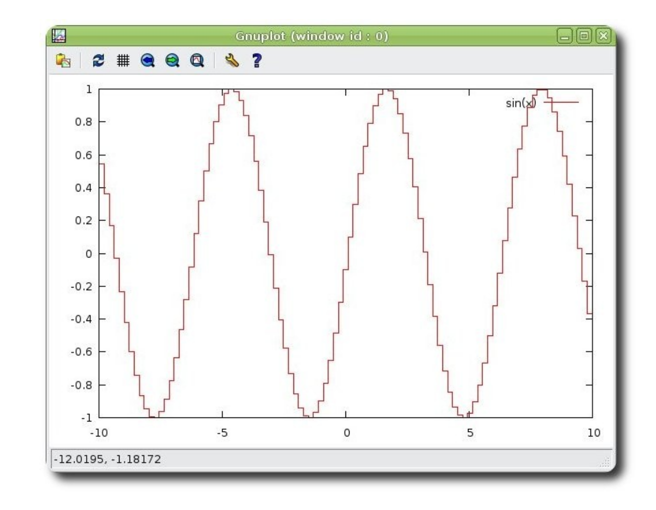
Estos estilos son asignados con la instrucción: **`plot sin(x) with impulses`**

```
plot sin(x) with impulses
```





```
plot sin(x) with steps
```

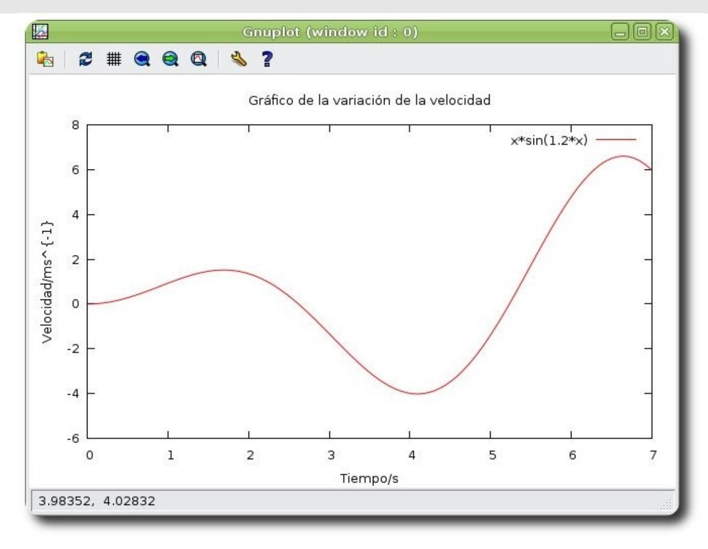


Personalizar las gráficas

Instrucción	Descripción
set title "Force-Deflection Data"	Asignar un título
set xlabel "Deflection (meters)"	Asignar una etiqueta en el eje X
set ylabel "Force (kN)"	Asignar una etiqueta en el eje Y
set xrange [0.001:0.005]	Cambiar el rango en el eje X
set yrange [20:500]	Cambiar el rango en el eje Y
set autoscale	Dejar que Gnuplot establezca el rango de los ejes

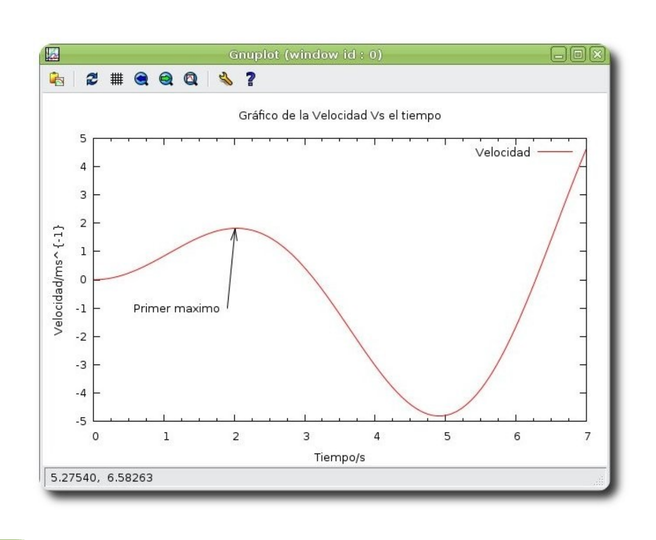
set key 0.01,100	Mover el identificador de la gráfica a la coordenada x,y
unset key	Eliminar el identificador de la gráfica
set label "yield point" at 0.003, 260	Poner una etiqueta en una coordenada x,y
unset label	Eliminar todas las etiquetas
set logscale	Dibujar usando ejes logarítmicos
unset logscale; set logscale y	Activar y desactivar escala logarítmica
set xtics (0.002,0.004,0.006,0.008)	
unset xtics; set xtics auto	

```
set title 'Gráfico de la variación de la velocidad'
set xlabel 'Tiempo/s'
set ylabel 'Velocidad/ms^{-1}'
set xrange[0:7]
plot x*sin(1.2*x)
```



Incluir indicadores

```
set title 'Gráfico de la Velocidad Vs el
tiempo'
set xlabel 'Tiempo/s'
set ylabel 'Velocidad/ms^{-1}'
set xrange[0:7]
set mxtics 4
set arrow 1 from  1.9,-1.0 to 2.01,1.8
set label 1 "Primer maximo" at 1.8,-1.0
right
plot x*sin(x) title "Velocidad"
```



Ajuste de curvas

Muchas veces surge la necesidad de poder ajustar nuestros datos, bajo un cierto criterio (lineal, no lineal, etc), una de las ventajas de usar **GNUPlot** va referida al ajuste de curvas dados los datos y el tiempo de ajuste que se precisa, aspecto que facilita enormemente el trabajo y la facilidad para los usuarios.

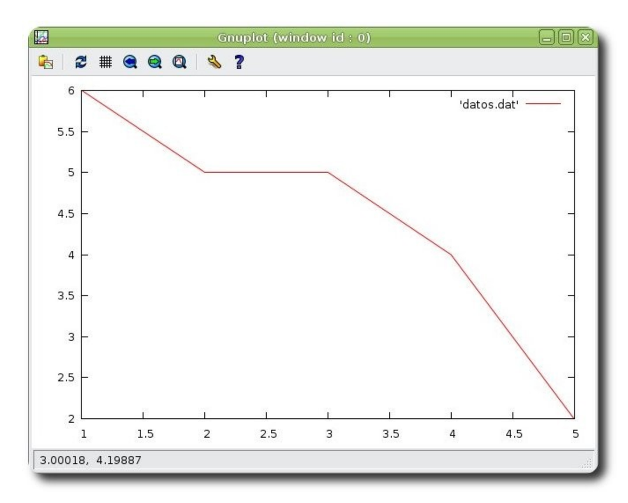
Inicialmente debemos considerar la presencia de un archivo que contenga los datos que serán ajustados, como se muestra en el ejemplo siguiente:

datos.dat

```
1       6
2       5
3       5
4       4
5       2
```

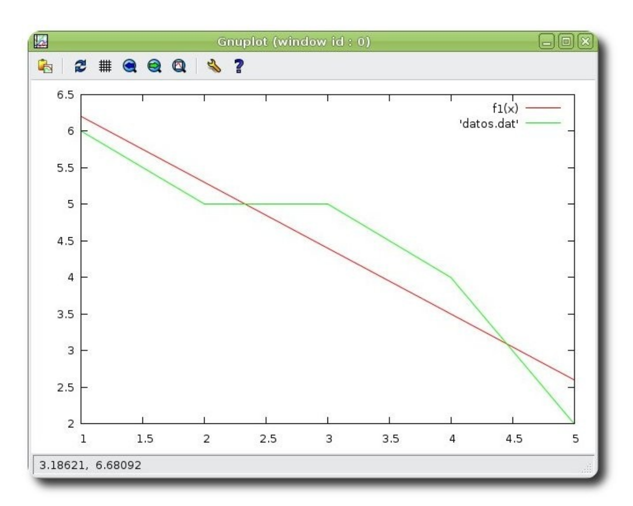
Realzamos el ploteo de los datos originales, para luego contrastar con los datos ajustados según diversas curvas.

```
plot "datos.dat" w l
```



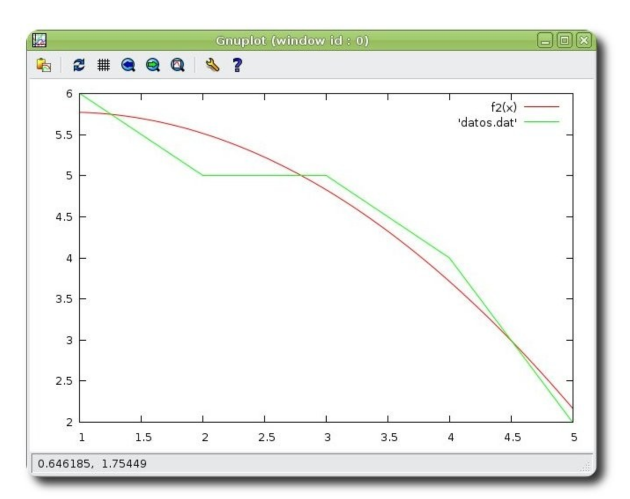
Ajuste a una ecuación lineal

```
f1(x)=a*x+b
fit f1(x) 'datos.dat' via a, b
# a=-0.9, b=7.1
plot f1(x) w l, 'datos.dat' w l
```



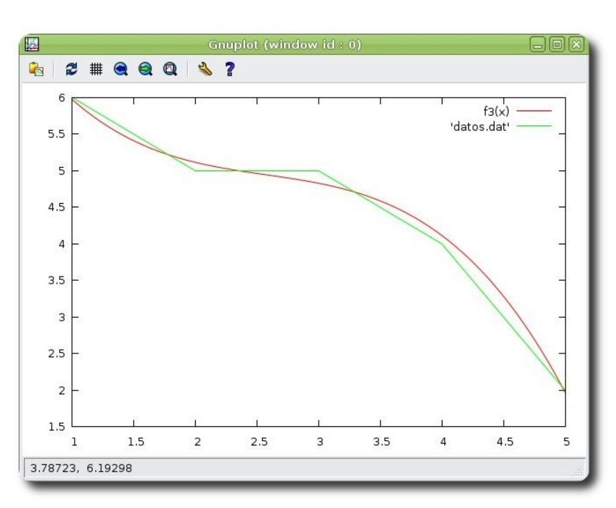
Ajuste a una ecuación polinomial de 2do grado

```
f2(x)=a*(x**2)+b*x+c
fit f2(x) 'datos.dat' via a, b, c
plot f2(x) w l, 'datos.dat' w l
```



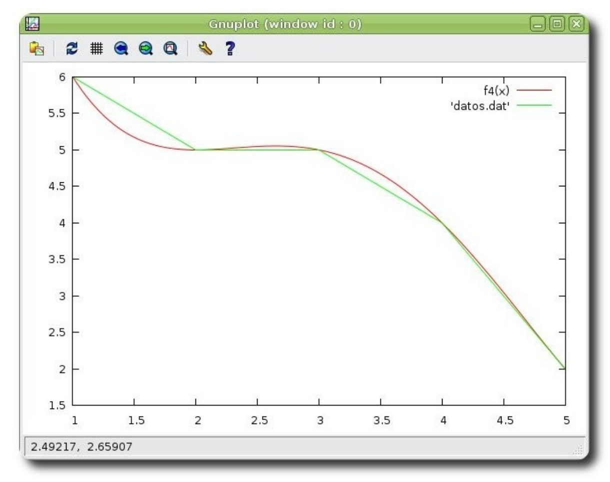
Ajuste a una ecuación polinomial de 3er grado

```
f3(x)=a*(x**3)+b*(x**2)+c*x+d
fit f3(x) 'datos.dat' via a, b, c, d
plot f3(x) w l, 'datos.dat' w l
```



Ajuste a una ecuación polinomial de 4to grado

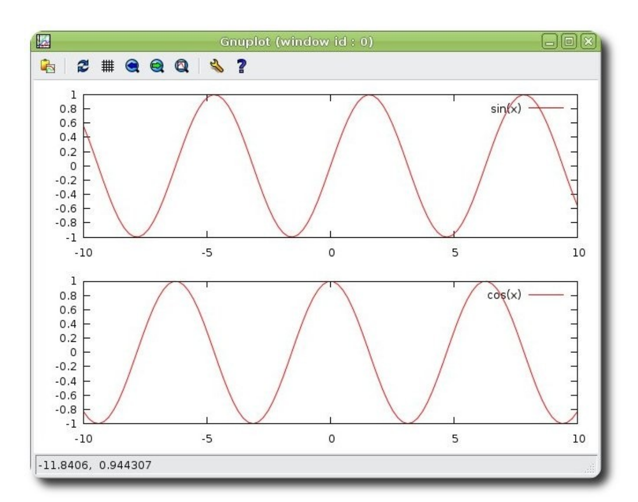
```
f4(x)=a*(x**4)+b*(x**3)+c*(x**2)+d*x+e
fit f4(x) 'datos.dat' via a, b, c, d, e
plot f4(x) w l, 'datos.dat' w l
```



Gráficas múltiples

GNUPlot permite realizar más de una gráfica en cada una de las salidas, permitiendo de esta forma realizar comparativas entre las distintas gráficas realizadas.

```
set multiplot;                                    #
get into multiplot mode
set size 1,0.5;
set origin 0.0,0.5;   plot sin(x);
set origin 0.0,0.0;   plot cos(x)
unset multiplot                                   #
exit multiplot mode
```



Grabar trabajos

Generalmente cuando estamos realizando trabajos con cierto grado de complejidad, el cual vamos ajustando paulatinamente hasta lograr los resultados esperados, es conveniente disponer de la posibilidad de poder grabar la secuencia de instrucciones realizadas para tal efecto, para este caso **GNUPlot** dispone de la instrucción **`save`**, y la **`load`** para recuperar la secuencia de instrucciones desde un archivo previamente guardado.

A continuación mostramos algunos ejemplos de su uso:

```
save 'trabajo.gnu'
save functions 'funciones.dat'
save var 'variables.dat'
save set 'opciones.dat'
save term 'terminal.gnu'
save '-'
save '|grep title >titulo.gp'
```

```
load 'trabajo.gnu'
load "funciones.dat"
load "< loadfile_generador.sh"
```

Salida impresa y gráfica

GNUPlot provee la posibilidad de direccionar la salida de sus gráficas, estas pueden ser redirigidas para ser impresas (formato PostScript) o pueden ser redirigidas hacia un archivo de imagen (gif, tgiff, jpeg, pbm, pdf, latex, pstricks, postscripts etc...) o la terminal X11 (opción por defecto)

Su uso podría ser:

```
set terminal gif
set output 'archivo01.gif'
plot....#<como usualmente se realiza>
set output  #puede ser necesario para
algunas terminales
```

Un ejemplo práctico con una salida a un archivo gif:

```
set terminal gif
set output 'archivo01.gif'
set title 'Gráfico de la Velocidad Vs el
tiempo'
set xlabel 'Tiempo/s'
set ylabel 'Velocidad/ms^{-1}'
set xrange[0:7]
set mxtics 4
set arrow 1 from  1.9,-1.0 to 2.01,1.8
set label 1 "Primer maximo" at 1.8,-1.0
right
plot x*sin(x) title "Velocidad"
```

Un ejemplo práctico con una salida a un archivo eps:

```
set xlabel 'time in seconds'
set ylabel 'throughput packets/second'
set yrange [15:25]
set term postscript enhanced color eps
set output 'grafico.eps'
plot 'ejemplo.dat' using 1:2 t
'throughput (with RTS)' w l,
'ejemplo.dat' using 1:3 t 'throughput
(without RTS)' w l
```

Interacción con lenguajes de programación

Una de las grandes virtudes de **GNUPlot** es brindar la posibilidad de ser utilizado desde una aplicación desarrollada en lenguajes de programación como Python, Perl, C, C++; cualidad que hace que se pueda reutilizar esta poderosa herramienta desde nuestras propias aplicaciones.

Poder de GNUPlot

Este artículo solo intenta mostrar una pequeÒa parte del gran potencial que se dispone en **GNUPlot**. Adicionalmente comentar que muchísimas universidades en todo el mundo han incluido a esta herramienta por sus buenas características, innumerables funciones implementadas y sobre todo por su facilidad de uso y rapidez en su curva de aprendizaje.

Referencias

[1] http://www.gnuplot.info/

Autores



Pilar Castel Diaz
Centro de Investigación en Física y Matemáticas (ES)
pilar.castel.diaz@gmail.com



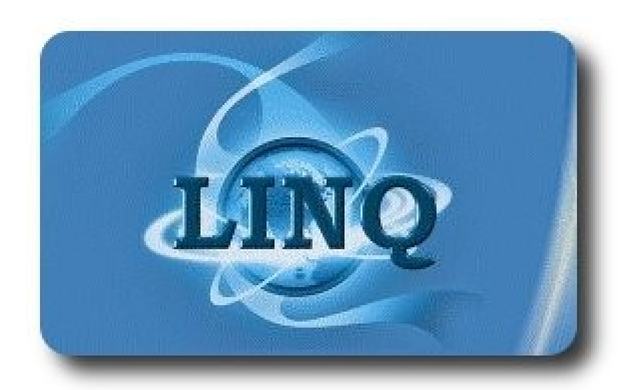


Esteban Saavedra López
Presidente de la Fundación AtixLibre (Bolivia)
Activista de Software Libre en Bolivia
jesaavedra@opentelematics.org
esteban.saavedra@atixlibre.org
http://jesaavedra.opentelematics.org
http://www.atixlibre.org



Entendiendo LINQ con C# y Mono

LINQ es una tecnología integrada en .NET que proporciona la capacidad para consultar o manipular diversas fuentes de datos, independientes del proveedor, utilizando de forma nativa la sintaxis de cualquier lenguaje de programación soportado por .NET



Introducción

Hoy en día es común encontrar aplicaciones en las cuales hay una necesidad de reemplazar o migrar de una o de varias fuentes de datos hacia otra u otras fuentes de datos similares o totalmente diferentes, si bien este cambio se da por razones de desempeño, económicas o de capacidad, este cambio implica que el desarrollador ajuste o reescriba el lenguaje de acceso a la fuente de datos (SQL, Macros,DOM, Xquery) para poner en marcha la aplicación utilizando la fuente de datos actualizada.

Un escenario así presenta las siguientes dificultades:

Los errores de sintaxis del lenguaje de acceso a la fuente de datos no son localizables por el compilador y solo se encuentran en tiempo de ejecución.

Obliga a que el desarrollador entienda a determinado nivel el lenguaje de manipulación de la fuente de datos, por ejemplo si es una base de datos se debe entender SQL, si es un XML entender DOM o XQuery.

Es inevitable la mezcla de lenguajes en la solución, para el desarrollo un lenguaje de programación imperativo orientado a objetos (como C#) y un lenguaje imperativo (como SQL) para la manipulación de datos.

En este escenario LINQ proporciona una solución estupenda, ya que las consultas estarían integradas dentro del lenguaje de programación sin importar cual fuese la fuente de datos y con la consistencia de utilizar el mismo patrón para todas las consultas.

Acerca de LINQ

Language Integrated Query o LINQ es una tecnología integrada en .NET que proporciona la capacidad para consultar o manipular diversas fuentes de datos, independientes del proveedor, utilizando de forma nativa la sintaxis de cualquier lenguaje de programación soportado por .NET, lo cual nos proporciona el soporte del compilador y nos permite concentrarnos únicamente en las búsquedas en lugar de cómo hacer la rutina para cada búsqueda, además la sintaxis de LINQ es similar a SQL lo que nos proporciona un estándar, ya que es la misma sintaxis para todas las fuentes de datos diferentes o similares.

Dependiendo de la fuente de datos a trabajar, es el componente LINQ a utilizar, los componentes se agrupan en:

- ✔ **LINQ to SQL**: Es el conjunto de clases, estructuras, interfaces y enumeraciones utilizadas para escribir consultas a bases de datos relacionales como PostgreSQL, SQL Server o MySQL.
- ✔ **LINQ to Objects**: Es la API predeterminada de LINQ y permite

escribir consultas para arreglos, estructuras y colecciones de objetos en memoria.

- **LINQ to XML**: Proporciona la habilidad de escribir consultas para procesar fuentes de datos XML.
- **LINQ to DataSet**: Es la API dedicada a trabajar con clases DataSets y DataTables, ya que aun existen aplicaciones y desarrolladores que utilizan esta solución.

En este tutorial mostraremos una serie de ejemplos con cada una de las expresiones y operadores de consulta de LINQ, utilizando el API predeterminada de LINQ o sea LINQ to Objects.

Trabajando LINQ con Mono

Antes de empezar a trabajar debemos tener instalado y configurado la última versión del proyecto Mono, ésto se consigue desde el sitio de descarga: **http://www.mono-project.com/Main_Page**

Empezaremos creando una aplicación de consola, para ejemplificar sobre todo las expresiones de consulta que se utilizan con LINQ.

Esta aplicación tendrá tres colecciones de objetos (entidades) Hospital, Almacén y Artículos la relación entre un hospital, un almacén y los artículos está dada por una relación de uno a muchos, donde un hospital tiene desde uno a varios almacenes y en cada almacén hay uno o varios artículos diferentes.

Para empezar con el ejemplo, escribiremos el código de las objetos que utilizaremos en el ejemplo, este código es el siguiente:

```
class Hospital {
 public int ID{set;get;}
 public string NombreH {set;get;}
}
class Almacen {
 public Hospital Hospital {set;get;}
 public int IDAlmacen{ set;get;}
 public string NombreA { set; get;}
}
class Articulo{
 public Almacen Almacen {set;get;}
 public int IDArticulo {set;get;}
 public string Nombre {set;get;}
 public double Precio{set;get;}
}
```

Ahora dentro del método Main escribimos el siguiente código, para iniciar las colecciones con datos donde realizaremos las búsquedas con LINQ.

```
List<Hospital> lHospitales = new List<Hospital>();
       var hospitales = new Hospital[] {
       new Hospital{ID = 1,NombreH="Angeles Pedregal"},
       new Hospital{ID = 2, NombreH="1ero Octubre"},
       new Hospital{ID = 3, NombreH="Ignacio Zaragoza"},
       new Hospital{ID = 4, NombreH="Angeles Torreon"}
       };
       List<Almacen> lAlmacenes = new List<Almacen>();
       var almacenes = new Almacen[]{
       new Almacen{Hospital = hospitales[0],IDAlmacen = 1,NombreA = "Patriotismo"},
       new Almacen{ Hospital = hospitales[0],IDAlmacen = 2,NombreA = "Zaragoza"},
       new Almacen{ Hospital = hospitales[1],IDAlmacen = 3,NombreA = "101"},
       new Almacen{ Hospital = hospitales[2],IDAlmacen = 4,NombreA = "404"},
       new Almacen{ Hospital = hospitales[1],IDAlmacen = 5,NombreA = "Monterrey"},
       new Almacen{ Hospital = hospitales[2],IDAlmacen = 6,NombreA = "Guadalajara"}
       };
       List<Articulo> lArticulos = new List<Articulo>();
       var articulos = new Articulo[]{
       new Articulo{Almacen = almacenes[0],IDArticulo = 1005,Nombre = "HOJA SIERRA 
18MM",Precio = 1.5},
       new Articulo{Almacen = almacenes[0],IDArticulo = 1006,Nombre = "DRENAJE BLAKE 
2229 15FR C/PUNZON°",Precio = 1.5},
       new Articulo{Almacen = almacenes[1],IDArticulo = 1007,Nombre = "VICRYL ETHICON 
3-0",Precio = 6.22},
       new Articulo{Almacen = almacenes[1],IDArticulo = 1008,Nombre = "PDS II* 
VIOLETA / ETHICON",Precio = 9.2},
```

```
        new Articulo{Almacen = almacenes[1],IDArticulo = 1009,Nombre = "IOBAN 2 
ANTIMICROBIAL / 3M",Precio = 3.11},
        new Articulo{Almacen = almacenes[2],IDArticulo = 1010,Nombre = "COMP FEM ANAT R/
C IZQ NO 1",Precio = 10.20},
        new Articulo{Almacen = almacenes[3],IDArticulo = 1011,Nombre = "BASE TIBIAL ANAT 
DER NO 7",Precio = 10.45},
        new Articulo{Almacen = almacenes[4],IDArticulo = 1012,Nombre = "INSERT ART UNIV 
UHMPE 3-4 X 9MM",Precio = 12.80},
        };
        //Agregamos la información
        foreach(Almacen a in almacenes)
        lAlmacenes.Add(a);
        foreach(Articulo r in articulos)
        lArticulos.Add(r);
```

Uso de la palabra reservada var

La palabra reservada var se utiliza para lograr una escritura más compacta en el código ya que le indica al compilador que infiera el tipo de la variable que se le asignará al declararle el valor inicial, por ejemplo en la forma clásica donde se declara el tipo de variable de forma explícita se escribe:

```
string s = "Revista Atix";
int i = 14;
DataSet ds = new DataSet();
```

utilizando la escritura de forma implícita con var se escribe:

```
var s = "Revista Atix";
var i = 14;
var ds = new DataSet();
```

El uso de la palabra reservada var tiene las siguientes limitaciones:

- ✔ siempre se debe asignar un valor inicial a la variable al declararse
- ✔ no se puede asignar a var un valor null
- ✔ no se puede asignar más de una variable var en una sentencia.

Las variables var solo tienen alcance local no pueden utilizarse como variables globales de clase.

Sintaxis y operadores de las consultas en LINQ

Básicamente la sintaxis de las consultas en LINQ es similar a las consultas SQL, utilizando las palabras reservadas select , from , where , orderby join.

Aquí a diferencia de las consultas SQL la palabra reservada **from** va al comienzo de la consulta y al final se emplea la palabra reservada **select** , como en el código siguiente:

```
var fromQuery = from a in lAlmacenes 
select a;
```

Similar a SQL tenemos la palabra reservada **where** , como en el código siguiente:

```
var whereQuery = from a in lAlmacenes 
where a.Hospital.ID == 2 select a;
```

El uso de **orderby** y **descending** se ejemplifica en los códigos siguientes:

```
var orderbyQuery = from a in lAlmacenes 
orderby a.NombreA select a;
var orderbyQueryDesc = from a in 
lAlmacenes orderby a.NombreA descending 
select a;
```

Continuando con la similitud con SQL no debe faltar la palabra reservada **join**, como en el siguiente código:

```
var joinQuery = from r in lArticulos join 
a in lAlmacenes on r.Almacen.IDAlmacen 
equals a.IDAlmacen select new 
{r.Nombre,a.NombreA };
```

Cada una de las consultas que se realizan con LINQ, regresan una interfaz **IEnumerable** , por eso en el método Display El parámetro que recibe es un **Ienumerable**, el código completo del ejemplo se lista a continuación:

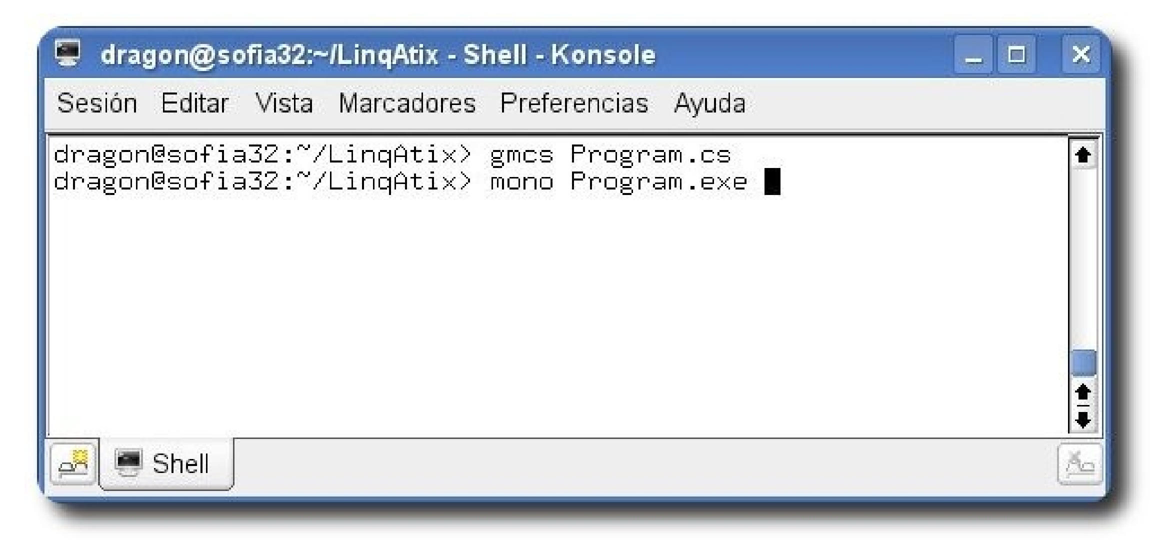
```
using System;
using System.Collections.Generic;
using System.Linq;

namespace LinqAtix
{
class Program{
public static void Main(string[] args){
       var hospitales = new Hospital[] {
       new Hospital{ID = 1,NombreH="Angeles Pedregal"},
       new Hospital{ID = 2, NombreH="1ero Octubre"},
       new Hospital{ID = 3, NombreH="Ignacio Zaragoza"},
       new Hospital{ID = 4, NombreH="Angeles Torreon"}
       };
       List<Almacen> lAlmacenes = new List<Almacen>();
       var almacenes = new Almacen[]{
       new Almacen{Hospital = hospitales[0],IDAlmacen = 1,NombreA = "Patriotismo"},
       new Almacen{ Hospital = hospitales[0],IDAlmacen = 2,NombreA = "Zaragoza"},
       new Almacen{ Hospital = hospitales[1],IDAlmacen = 3,NombreA = "101"},
       new Almacen{ Hospital = hospitales[2],IDAlmacen = 4,NombreA = "404"},
       new Almacen{ Hospital = hospitales[1],IDAlmacen = 5,NombreA = "Monterrey"},
       new Almacen{ Hospital = hospitales[2],IDAlmacen = 6,NombreA = "Guadalajara"}
       };
       List<Articulo> lArticulos = new List<Articulo>();
       var articulos = new Articulo[]{
       new Articulo{Almacen = almacenes[0],IDArticulo = 1005,Nombre = "HOJA SIERRA
18MM",Precio = 1.5},
       new Articulo{Almacen = almacenes[0],IDArticulo = 1006,Nombre = "DRENAJE BLAKE
2229 15FR C/PUNZON°",Precio = 1.5},
       new Articulo{Almacen = almacenes[1],IDArticulo = 1007,Nombre = "VICRYL ETHICON
3-0",Precio = 6.22},
       new Articulo{Almacen = almacenes[1],IDArticulo = 1008,Nombre = "PDS II*
VIOLETA / ETHICON",Precio = 9.2},
       new Articulo{Almacen = almacenes[1],IDArticulo = 1009,Nombre = "IOBAN 2
ANTIMICROBIAL / 3M",Precio = 3.11},
       new Articulo{Almacen = almacenes[2],IDArticulo = 1010,Nombre = "COMP FEM ANAT R/
C IZQ NO 1",Precio = 10.20},
       new Articulo{Almacen = almacenes[3],IDArticulo = 1011,Nombre = "BASE TIBIAL ANAT
DER NO 7",Precio = 10.45},
       new Articulo{Almacen = almacenes[4],IDArticulo = 1012,Nombre = "INSERT ART UNIV
UHMPE 3-4 X 9MM",Precio = 12.80},
       };
       //Agregamos la información
       foreach(Almacen a in almacenes)
       lAlmacenes.Add(a);
       foreach(Articulo r in articulos)
       lArticulos.Add(r);
       //Consultas Linq:Trabajando con FROM
       var fromQuery = from a in lAlmacenes select a;
       //listamos todos los almacenes
       Display(fromQuery);
       //listamos todos los almacenes donde el id Hospital sea el 2
       var whereQuery = from a in lAlmacenes where a.Hospital.ID == 2 select a;
       Display(whereQuery);
       //listamos todos los almacenes en orden alfabetico
       var orderbyQuery = from a in lAlmacenes orderby a.NombreA select a;
       Display(orderbyQuery);
```

```
        //listamos todos los almacenes en orden alfabetico en forma descendente
        var orderbyQueryDesc = from a in lAlmacenes orderby a.NombreA descending select
a;
        Display(orderbyQueryDesc);
        //listamos el nombre de los articulos y el nombre del Hospital
        var joinQuery = from r in lArticulos join a in lAlmacenes on r.Almacen.IDAlmacen
equals a.IDAlmacen select new {r.Nombre,a.NombreA };
        Console.WriteLine("======Resultados del Query======");
        foreach(var j in joinQuery)
        Console.WriteLine("Articulo: {0}\tAlmacén: {1}",j.Nombre,j.NombreA);
        Console.ReadKey(true);
        }
        static void Display(IEnumerable<Almacen> ie){
        Console.WriteLine("======Resultados del Query======");
        foreach(Almacen a in ie)
        Console.WriteLine("{0}\t{1}",a.IDAlmacen,a.NombreA);
        }
        }

        class Hospital {
        public int ID{set;get;}
        public string NombreH {set;get;}
        }
        class Almacen {
        public Hospital Hospital {set;get;}
        public int IDAlmacen{ set;get;}
        public string NombreA { set; get;}
        }
        class Articulo{
        public Almacen Almacen {set;get;}
        public int IDArticulo {set;get;}
        public string Nombre {set;get;}
        public double Precio{set;get;}
        }
}
```

Para compilar el ejemplo, debemos de utilizar el comando gmcs, como se muestra en la imagen siguiente:



Para ejecutar el programa utilizamos el comando mono **`Program.exe`**, y la salida correspondiente se muestra en la imagen siguiente:

```
dragon@sofia32:~/LinqAtix - Shell - Konsole
Sesión  Editar  Vista  Marcadores  Preferencias  Ayuda
5       Monterrey
6       Guadalajara
======Resultados del Query======
3       101
5       Monterrey
======Resultados del Query======
3       101
4       404
6       Guadalajara
5       Monterrey
1       Patriotismo
2       Zaragoza
======Resultados del Query======
2       Zaragoza
1       Patriotismo
5       Monterrey
6       Guadalajara
4       404
3       101
======Resultados del Query======
Articulo: HOJA SIERRA 18MM      Almacén: Patriotismo
Articulo: DRENAJE BLAKE 2229 15FR C/PUNZONº      Almacén: Patriotismo
Articulo: VICRYL ETHICON 3-0    Almacén: Zaragoza
Articulo: PDS II* VIOLETA / ETHICON     Almacén: Zaragoza
Articulo: IOBAN 2 ANTIMICROBIAL / 3M    Almacén: Zaragoza
Articulo: COMP FEM ANAT R/C IZQ NO 1    Almacén: 101
Articulo: BASE TIBIAL ANAT DER NO 7     Almacén: 404
Articulo: INSERT ART UNIV UHMPE 3-4 X 9MM       Almacén: Monterrey
Shell
```

Conclusión

LINQ es una más de las muchas tecnologías que ofrece .NET y Mono para facilitar el trabajo a los desarrolladores en cuanto al trabajo con fuentes de datos, LINQ es un enfoque total de llevar la manipulación de datos en los lenguajes orientados a objetos, librando los inconvenientes de trabajar con los lenguajes de manipulación de datos que cada proveedor posee.

Los ejemplos pueden ser descargados de **`http://www.humansharp.com/`**

Autor



I.S. Martín Márquez
xomalli@gmail.com

SQLite El gestor de bases de datos ultra ligero. (2da Parte)

Nuestro objetivo, crear una pequeña aplicación en lenguaje Java que sea capaz de ilustrar la combinación de dos grandes tecnologías para trabajar con bases de datos.



Introducción

Nuevamente aquí, para tratar la continuación de SQLite como alternativa ligera a los grandes Sistemas gestores de bases de datos. En la primera parte de este tema nos concentramos en describir las características, ventajas y desventajas, cuando es y cuando no es recomendable utilizar esta herramienta, entre otros. En esta segunda parte nos enfocaremos más a un caso práctico donde se ejemplifica de forma sencilla el acceso a bases de datos SQLite desde Java.

El hecho de utilizar Java para este ejemplo práctico radica principalmente en que esta plataforma de desarrollo se ha popularizado mucho en los últimos años, tanto en los medios universitarios como en el ámbito laboral, Java es famoso.

El proyecto

Sea la siguiente situación problema:

“En una tienda en línea que vende libros, tanto impresos como electrónicos (ebook), se tiene la necesidad de llevar un control de la información de cada uno de los libros que la empresa vende, datos como: Nombre del libro, Autor, edición, ISBN, idioma en que está escrito el libro, y por último el precio. Se le pide a usted como programador la creación de una aplicación que permita llevar a cabo el control de estos datos.”

Como pueden ver lo que se nos pide no es muy complicado, con una base de datos, una única tabla en donde almacenar los campos correspondientes y listo. El punto interesante aquí es como poder administrar dicha información de forma externa al manejador de la base de datos, en nuestro caso desde una aplicación creada en lenguaje Java.

Estableciendo conexión Java - SQLite

Si echan un vistazo por la Web se darán cuenta de la importancia que a cobrado SQLite en el mundo de las bases de datos y por ende en la forma de trabajo de los desarrolladores de software, a tal grado que estos días existen una gran cantidad de drivers o wrappers que facilitan la conexión a bases de datos SQLite con la mayoría de los lenguajes de programación más conocidos... y los no tanto.

Nosotros que estamos trabajando con Java, uno de los lenguajes más populares dentro del desarrollo de software contamos con una buena variedad de opciones que nos permiten establecer conexión entre nuestras aplicaciones y las bases de datos SQLite.

A continuación algunas de las herramientas de conexión a bases de datos SQLite con Java:

Wrapper	Breve descripción	Link
javasqlite	Un driver JDBC para SQLite v2 y v3.	http://www.ch-werner.de/javasqlite
Sqlite wrapper for java	Java wrapper desarrollado en torno a SWIG(conecta aplicaciones C/C++ a otros lenguajes).	http://tk-software.home.comcast.net/
SQLiteJDBC	JDBC driver para Windows, Linux, Mac OS X, y también una versión 100% Java.	http://www.zentus.com/sqlitejdbc
sqlite-java-shell	UN port 100% Java puro de la línea de comandos SQLite3 escrito con NestedVM. (No es un driver JDBC).	http://www.sqlite.org/contrib?orderby=date
SQLite JDBC Driver for Mysaifu JVM	SQLite JDBC Driver para Mysaifu JVM (JVM para Windows Mobile) y también para las bibliotecas SQLite JNI(Java Native Interface) para Windows (x86), Linux (i386/PowerPC) y Mac OS X (Intel).	http://sourceforge.jp/projects/sqlitejdbc/
Xerial: SQLite JDBC Driver	Driver JDBC para SQLite que contiene versiones compiladas nativas para sistemas Windows,(x86), Mac OS X(intel i386) y Linux(i386, amd64). Otros sistemas también son soportados con la versión SQLite 100% Java.	http://www.xerial.org/trac/Xerial/wiki/SQLiteJDBC
SQLJet	Una implementacion 100% Java de SQLite(no utiliza NestedVM).	http://www.sqljet.com

Tabla 1. Driver SQLite para Java.

Estas son algunas de las herramientas más conocidas para la conexión de bases de datos SQLite con Java, (existen algunas más). Para mayor información pueden consultar lo necesario en los sitios Web pertinentes.

Requerimientos



Gráfico 2. Nuestra elección para enlazar java y SQLite.

De las opciones que se presentan voy ha decantar la utilización de Xerial SQLite JDBC Driver, el cual es una extensión de Zentus's SQLite JDBC Driver, debido a la facilidad de uso. La verdad es que no he probado todas estas herramientas, no puedo asegurar que mi elección sea la mejor pero para nuestro propósito me parece una buena opción.

Para el ejemplo que trabajaremos es necesario descargar el archivo sqlite-jdbc-xx.yy.zz.jar de la página de oficial de Xerial SQLite JDBC Driver y colocar dicho archivo en la carpeta de trabajo donde se encuentren los códigos.

Además está claro que necesitamos tener instalado SQLite en nuestro ordenador, para el ejemplo utilizamos SQLite 3.

Estructura de nuestra aplicación

Para nuestro ejemplo haremos un pequeño programa que nos permita realizar básicamente 3 operaciones; inserción de registros, consultas y eliminación de registros. A continuación se detalla el código más relevante de la aplicación, clases a utilizar y los métodos para cada una de las operaciones.

El driver que vamos a usar es un driver JDBC, que éste trabaja con las bibliotecas de clases que java dispone para la conexión a bases de datos, dichas clases se encuentran contenidas en el paquete java.sql el cual será necesario incluir en nuestro programa, todo o solo las clases que necesitemos. En nuestro ejemplo solo utilizaremos las siguientes:

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
```

Listado 1. Clases necesarias para la conexión.

La función de cada una de estas clases se verá a continuación.

Inserción de Registros

La primera operación es insertar registros en la nueva base de datos solicitando al usuario que introduzca datos, nos apoyamos de la clase JOptionPane para facilitar la introducción de los datos, para nuestro ejemplo damos por hecho que el usuario introducirá datos correctos, controlar todos esos detalles toma muchas líneas de código, en este ejemplo no lo haremos; para aplicaciones más formales el control de la entrada de datos es necesaria o más bien dicho es OBLIGATORIA.

```
1	public void insertaRegistro()
2	{
3	    String nombreLibro = "", autor = "", isbn = "", idioma = "";
4	    int edicion = 1;
5	    float precio = 0f;
6
7	    try {
8	        nombreLibro = JOptionPane.showInputDialog("Nombre del libro: ");
9	        autor  = JOptionPane.showInputDialog("Autor del libro: ");
10	        edicion = Integer.parseInt(JOptionPane.showInputDialog("Edicion:
	"));
11	        isbn = JOptionPane.showInputDialog("ISBN: ", "Nuevo registro");
12	        idioma = JOptionPane.showInputDialog("Idioma: ");
13	        precio = Float.parseFloat(JOptionPane.showInputDialog("Precio:
	"));
14
15	    } catch(Exception e) { System.out.println("ERROR - " + e.toString());
16	}
17
18	    try {
19	        sentencia = conexion.createStatement();
20	        sentencia.setQueryTimeout(30);
21	        sentencia.executeUpdate("INSERT INTO libros VALUES('" +
	nombreLibro + "', '" + autor + "', " + edicion + ", '" + isbn + "', '" +
	idioma + "', " + precio + ")");
22
23	    }catch(SQLException sqle){ System.out.println("ERROR - " +
	sqle.toString()); }
24	}//Fin insertaRegistro()
```

Listado 2. Método para inserción de registros.

Veamos brevemente la función de cada una de las líneas resaltadas. En la línea 18 preparamos una sentencia para aplicarse sobre la base de datos a la cual apunta conexion, en la línea 19 establecemos un tiempo de espera para cada transacción, de la línea 20 a la 22 procedemos a ejecutar una operación sobre la base de datos, insertamos un nuevo registro.

Consultas

Una de las operaciones más importantes de las bases de datos es sin duda la recuperación de datos, con este código se realiza una consulta que nos permite recuperar información de la base de datos y la muestra al usuario de forma ordenada.

```
public void realizaConsulta()
{
    String clave = "", resultados = "";
    clave = JOptionPane.showInputDialog("Dato clave para la consulta " +
                                        " (en blanco para todo): ");
    try{
        String consulta = "SELECT * FROM libros WHERE nombre = '" + clave+
                    "' or autor = '" + clave + "' or isbn = '" + clave + "'";
        if(clave.equals(""))     //Una consulta de todo
            consulta = "SELECT * FROM libros";



        sentencia = conexion.createStatement();
        sentencia.setQueryTimeout(30);

        ResultSet cs = sentencia.executeQuery(consulta);

```

```
17          while(cs.next()) {
18              resultados += "Libro:    " + cs.getString("nombre") + "\n";
19              resultados += "Autor:    " + cs.getString("autor") + "\n";
20              resultados += "Edicion:  " + cs.getInt("edicion") + "\n";
21              resultados += "ISBN:     " + cs.getString("isbn") + "\n";
22              resultados += "Idioma:   " + cs.getString("idioma") + "\n";
23              resultados += "Precio:  $" + cs.getFloat("precio") + "\n\n";
24          }
25          JTextArea texto = new JTextArea(resultados);
26          texto.setFont(new Font("Sans-Serif", Font.PLAIN, 10));
27          texto.setEditable(false);
28          JScrollPane scroll = new JScrollPane(texto);
29          scroll.setPreferredSize(new Dimension(250, 400));
30
31          JOptionPane.showMessageDialog(null, scroll);
32      }catch(SQLException sqle){ System.out.println("ERROR - " +
sqle.toString()); }
33
34  }//Fin realizaConsulta()
```

Listado 3. Método para realizar consultas.

Con este método procedemos a pedir datos al usuario para realizar la consulta y posteriormente en las líneas 7 y 8 formulamos la consulta en base a la información proveída por el usuario, en caso de que la cadena esté vacía, en las líneas 9 y 10 formulamos una consulta total.

En la línea 15 se crea un ejemplar de la clase ResultSet que permite almacenar los datos resultantes de una consulta, de las líneas 17 a 24 procedemos a dar formato a la salida de la consulta para poder presentarla de forma ordenada.

Eliminación de Registros

Si por algún motivo se necesita eliminar algún registro, o varios de ellos, con este método tenemos la posibilidad de hacerlo. La eliminación se basa en 4 campos como son nombre del libro, Autor o el ISBN, o en su caso la eliminación de todos los registros para una entrada de '*'. La forma en que se realiza la operación es muy tonta xD, pero de momento solo estamos experimentando, no es necesario tanto control.

```
public void eliminaRegistros()
{
    String clave = "";
    clave = JOptionPane.showInputDialog("Nombre de libro, Autor a eliminar o
ISBN" + " (* para eliminar todo): ");
    try{
        //Eliminamos solo en base al titulo del libro, nombre de autor o ISBN
        String comando = "DELETE FROM libros WHERE nombre = '" + clave +
                    "' or autor = '" + clave  + "' or isbn = '" + clave + "'";

        if(clave.equals("*"))    //Eliminar todos los registros
            comando = "DELETE FROM libros";

        sentencia = conexion.createStatement();
        sentencia.setQueryTimeout(30);
        sentencia.executeUpdate(comando);
    }catch(SQLException sqle) {  System.out.println("ERROR - " +
sqle.toString()); }
}
```

Listado 4. Método para la eliminación de registros.

Las sentencias que se utilizan en este método consisten básicamente en solicitar al usuario los

datos de un libro, para en base a eso poder eliminar los registros pertinentes, en caso de recibir como entrada el símbolo '*' el usuario indica que quiere eliminar todos los registros de la base de datos, líneas 11 y 12 controlan esta situación.

Los tres métodos anteriores contienen lo sustancial del trabajo con la base de datos, a continuación se lista el código completo para la mimi-aplicación que he preparado para este ejemplo, aparte del código antes listado lo demás es solo para preparar la interfaz de usuario.

```
import java.io.*; import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;

public class DatabaseConnection {
    private Connection conexion;
    private Statement sentencia;
    public DatabaseConnection(String databaseName) throws ClassNotFoundException
    {
        //Cargamos driver para SQLite
        Class.forName("org.sqlite.JDBC");
        conexion = null;
        try{
            //Creamos la base de datos de nombre indicado como parametro
            conexion = DriverManager.getConnection("jdbc:sqlite:"+databaseName+".db");
            sentencia = conexion.createStatement();
            sentencia.setQueryTimeout(30);
            //Creamos la tabla, si existe se sobreescribe
            sentencia.executeUpdate("DROP TABLE IF EXISTS libros");
            sentencia.executeUpdate("CREATE TABLE libros " +
                    "(nombre STRING, autor STRING,  edicion INTEGER, " +
                    "isbn INTEGER, idioma STRING, precio REAL )");

        }catch(SQLException sqle){ System.out.println("ERROR - " + sqle.toString()); }
    }

    public void insertaRegistro()
    {
        String nombreLibro = "", autor = "", isbn = "", idioma = "";
        int edicion = 1;
        float precio = 0f;

        try {
            nombreLibro = JOptionPane.showInputDialog("Nombre del libro: ");
            autor  = JOptionPane.showInputDialog("Autor del libro: ");
            edicion = Integer.parseInt(JOptionPane.showInputDialog("Edicion: "));
            isbn = JOptionPane.showInputDialog("ISBN: ", "Nuevo registro");
            idioma = JOptionPane.showInputDialog("Idioma: ");
            precio = Float.parseFloat(JOptionPane.showInputDialog("Precio: "));

        } catch(Exception e) { System.out.println("ERROR - " + e.toString()); }

        try {
            sentencia = conexion.createStatement();
            sentencia.setQueryTimeout(30);
            sentencia.executeUpdate("INSERT INTO libros VALUES('" + nombreLibro +
                                   "', '" + autor + "', " + edicion + ", '" + isbn +
                                    "', '" + idioma + "', " + precio + ")");

        }catch(SQLException sqle){ System.out.println("ERROR - " + sqle.toString()); }
    }//Fin insertaRegistro()
```

```
    public void realizaConsulta()
    {
        String clave = "", resultados = "";
        clave = JOptionPane.showInputDialog("Dato clave para la consulta " +
                                            " (en blanco para todo): ");
        try{
            String consulta = "SELECT * FROM libros WHERE nombre = '" + clave+
                            "' or autor = '" + clave + "' or isbn = '" + clave + "'";
            if(clave.equals("")) //Una consulta de todo
                consulta = "SELECT * FROM libros";

            sentencia = conexion.createStatement();
            sentencia.setQueryTimeout(30);

            ResultSet cs = sentencia.executeQuery(consulta);

            while(cs.next()) {
                resultados += "Libro:   " + cs.getString("nombre") + "\n";
                resultados += "Autor:   " + cs.getString("autor") + "\n";
                resultados += "Edicion: " + cs.getInt("edicion") + "\n";
                resultados += "ISBN:    " + cs.getString("isbn") + "\n";
                resultados += "Idioma:  " + cs.getString("idioma") + "\n";
                resultados += "Precio:  $" + cs.getFloat("precio") + "\n\n";
            }
            JTextArea texto = new JTextArea(resultados);
            texto.setFont(new Font("Sans-Serif", Font.PLAIN, 10));
            texto.setEditable(false);
            JScrollPane scroll = new JScrollPane(texto);
            scroll.setPreferredSize(new Dimension(250, 400));

            JOptionPane.showMessageDialog(null, scroll);
        }catch(SQLException sqle){ System.out.println("ERROR - " + sqle.toString()); }

    }//Fin realizaConsulta()

    public void eliminaRegistros()
    {
        String clave = "";
        clave = JOptionPane.showInputDialog("Nombre de libro, Autor a eliminar o " +
                                            "ISBN (* para eliminar todo): ");
        try{
            //Eliminamos solo en base al titulo del libro, nombre de autor o ISBN
            String comando = "DELETE FROM libros WHERE nombre = '" + clave +
                            "' or autor = '" + clave  + "' or isbn = '" + clave + "'";

            if(clave.equals("*"))   //Eliminar todos los registros
                comando = "DELETE FROM libros";

            sentencia = conexion.createStatement();
            sentencia.setQueryTimeout(30);
            sentencia.executeUpdate(comando);
        }catch(SQLException sqle) {
            System.out.println("ERROR - " + sqle.toString());
        }
    }
}
```

Listado 5. Clase DatabaseConnection que trabaja con la base de datos.

Ahora la clase que utiliza a DatabaseConnection y muestra la interfaz de usuario.

```
import java.io.*;
import java.sql.*;
import java.awt.*;
import java.awt.event.*;
import java.awt.FlowLayout;

import javax.swing.JFrame;
import javax.swing.JButton;
import javax.swing.JLabel;


public class AdminApp extends JFrame {
    //private static AdminApp ventana;
    private JLabel lblLeyenda;
    private JButton btnInsertar;
    private JButton btnConsultar;
    private JButton btnEliminar;
    private DatabaseConnection conexion;

    public AdminApp() throws IOException
    {
        super(".: Accesando a SQLite :.");
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setResizable(false);
        setLayout(new FlowLayout(FlowLayout.CENTER, 200, 10));

        lblLeyenda = new JLabel("<html>Eliga una operacion: <br>");
        btnInsertar = new JButton("Inserta Registro");
        btnConsultar = new JButton("Consulta datos");
        btnEliminar = new JButton("Elimina Registro");

        try{
            conexion = new DatabaseConnection("Ejemplo");
        }catch(Exception e){
            System.out.println("ERROR - " + e.toString());
        }

        ActionListener accion =  new ActionListener() {
            public void actionPerformed(ActionEvent event)
            {
                if(event.getSource() == btnInsertar)
                    conexion.insertaRegistro();
                else if(event.getSource() == btnConsultar)
                    conexion.realizaConsulta();
                else if(event.getSource() == btnEliminar)
                    conexion.eliminaRegistros();
            }
        };
        btnInsertar.addActionListener(accion);
        btnConsultar.addActionListener(accion);
        btnEliminar.addActionListener(accion);
        add(lblLeyenda);
        add(btnInsertar);
        add(btnConsultar);
        add(btnEliminar);
        setSize(350, 200);
    }
    public static void main(String[] argumentos)  throws IOException
    {
        AdminApp ventana = new AdminApp();
        ventana.setVisible(true);
    }
}
```

Listado 6. Clase principal AdminApp.

Compilación y resultado

Una vez teniendo el código anterior procedemos a compilarlo normalmente con:

```
$ javac AdminApp.java
```

Ahora, para ejecutar la aplicación es necesario contar con el archivo sqlite-jdbc-xx.yy.zz.jar (xx.yy.zz es la versión la cual puede variar ligeramente) el cual pueden descargar de la página oficial de Xerial SQLite JDBC Driver, si no lo ha hecho aun, ahora es buen tiempo para hacerlo. Una vez descargado el archivo lo colocamos en el directorio donde estén los códigos fuentes de la aplicación, también pueden colocarse en algún otro directorio solo que se tendría que especificar dicha ruta el el classpath para poder ejecutar el programa, en el ejemplo utilizamos el directorio actual.

```
$ java -cp .:sqlite-jdbc-3.6.19.jar AdminApp
```

Del comando anterior se ejecuta la aplicación utilizando bibliotecas nativas a la plataforma en la que se ejecute el programa. Xerial SQLite JDBC permite realizar la ejecución utilizando código 100% java, hay dos formas de hacerlo:

```
$ java -Dsqlite.purejava=true -cp .:sqlite-jdbc-3.6.19.jar AdminApp
```

O con setProperty() de la clase System antes de cargar el controlador de la base de datos, quedando de la siguiente manera:

```
System.setProperty("sqlite.purejava", "true");
Class.forName("org.sqlite.JDBC");
```

Ahora, si todo va bien el resultado es el siguiente:



Gráfico 3. Las 3 operaciones de nuestra aplicación.

La primera operación, la de inserción de registros se ve en acción al presionar el primer botón.



Gráfico 4. Insertando registros a la base de datos.

Después de haber agregado registros a la base de datos es posible que quieran consultar dichos datos, esto es posible con la segunda opción de nuestro programa.

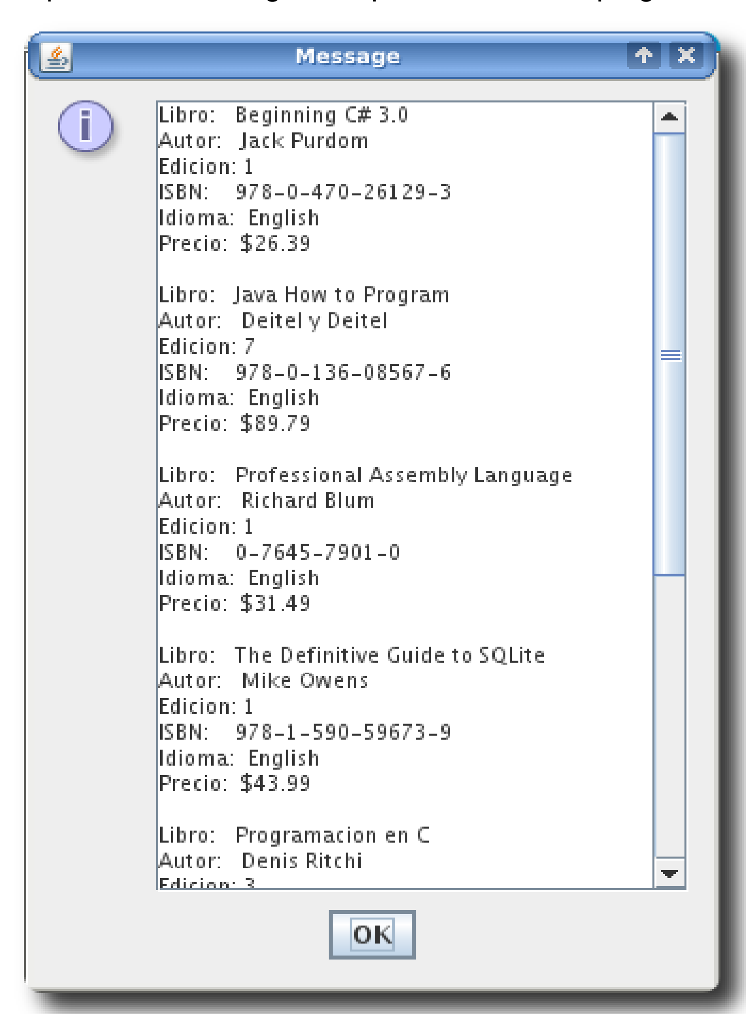


Gráfico 5. Resultado de la consulta a la base de datos.

Como última opción se tiene la posibilidad de eliminar registros de nuestra base de datos en base al nombre del libro, el autor, o en su caso el ISBN del libro o libros a eliminar. Se da la opción de eliminar la tabla entera con el símbolo '*' como entrada.

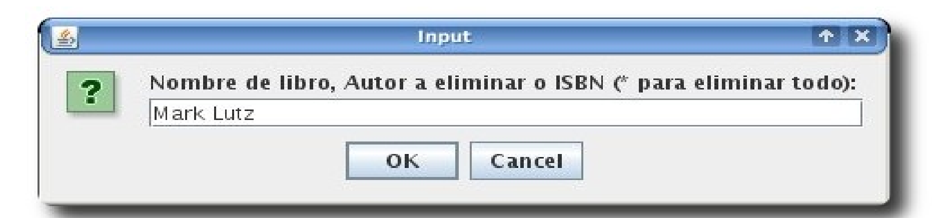


Gráfico 6. Eliminación de registros por autor.

Al parecer todo funciona, pero por si no se convencen o quieren estar bien seguros de que la aplicación hace lo que dice podemos monitorizar el contenido de nuestra base de datos con algunas de las aplicaciones gráficas que existen para SQLite, en la primera parte de este tema se listaron algunas de ellas. Veamos que pasa utilizando el SQLiteStudio para inspeccionar el contenido de nuestra base de datos.

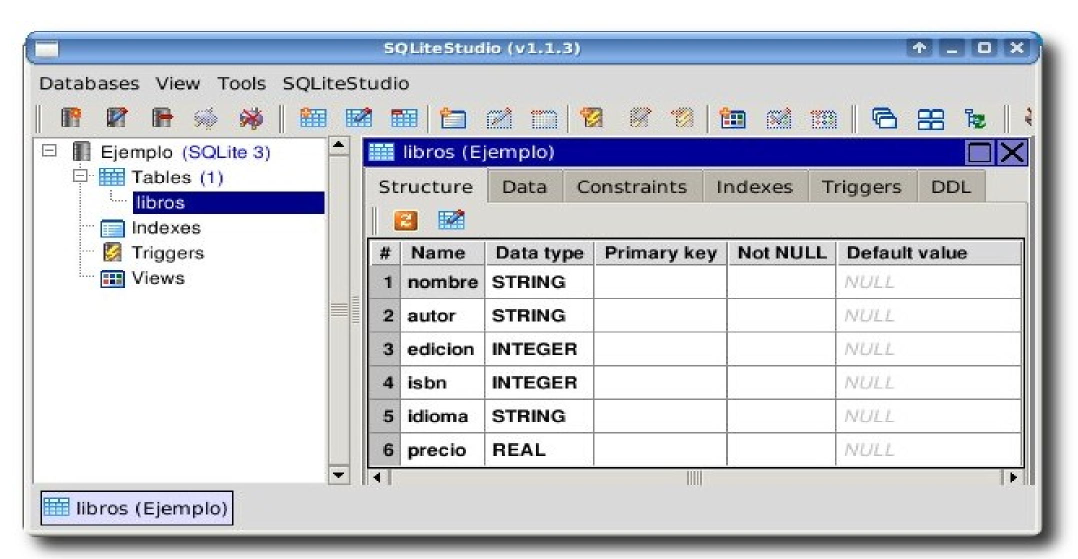


Gráfico 7. El mismo contenido que maneja nuestra aplicación pero desde SQLiteStudio.

Ahora veamos los datos almacenados.

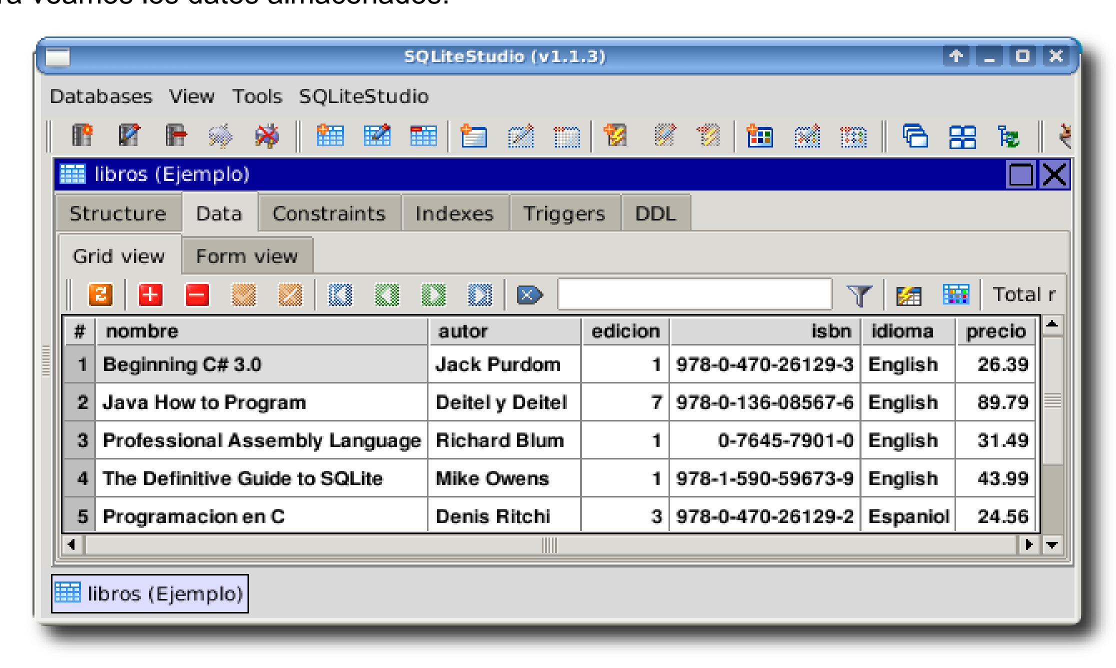


Gráfico 8. Los datos que hemos insertado en nuestra base de datos vistos desde SQLiteStudio.

El código de esta aplicación estará disponible en **`inforscience.260mb.com/data/SQLite.mas.Java.ATIX.zip`** para que les sea más fácil acceder al código.

Conclusión

Como se muestra en este documento existen varias herramientas que permiten a las bases de datos SQLite conectarse y operar con una gran variedad de lenguajes, siendo Java uno de los más privilegiados al contar con varias de ellas. Xerial JDBC SQLite Driver nos permite conectar SQLite con Java de forma sencilla tal y como se ha visto en este documento.

Referencias

[1] www.sqlite.org

[2] www.xerial.org

Autor



Rafael Rendón Pablo
Estudiante de Ingeniería en Sistemas Computacionales
Usuario de software libre
smart.dyr@gmail.com
www.inforscience.260mb.com



Monitoreo de acceso a páginas Web en línea

En el anterior número de esta revista vimos la forma de analizar los logs de nuestras páginas Web, éste es un tipo de monitoreo fuera de línea (offline), ahora presentaremos una forma de monitoreo en línea (online) de nuestras páginas Web utilizando la herramienta APACHETOP.



Introducción

El APACHETOP es una herramienta de monitoreo en tiempo real en modo texto, parecida a la herramienta TOP, con la cual podemos desplegar información de nuestro servidor Web APACHE como: el estado, estadísticas de acceso, etc.

Instalación de Apachetop

Se debe bajar el instalador de la URL **http://www.webta.org/projects/apachetop** Utilizaremos la distribución DEBIAN para implementar el APACHETOP..

Para la instalación ejecutaremos el siguiente comando:

```
# apt-get install apachetop
```

Ejecutando Apachetop

La forma básica de ejecutar la herramienta APACHETOP es ejecutando:

```
# apachetop -f <archivo de log de acceso>
```

Para este ejemplo:

```
# apachetop -f /var/log/apache2/access.log
```

```
Terminal
File  Edit  View  Terminal  Tabs  Help
last hit: 08:12:55        atop runtime:  0 days, 00:00:55          08:13:02
All:            10 reqs (   0.2/sec)        58.4K (  1425.0B/sec)   5985.1B/req
2xx:       10 ( 100%) 3xx:       0 (  0.0%) 4xx:      0 (  0.0%) 5xx:     0 (  0.0%)
R ( 30s):        3 reqs (   0.1/sec)        10.3K (   352.5B/sec)   3524.7B/req
2xx:        3 ( 100%) 3xx:       0 (  0.0%) 4xx:      0 (  0.0%) 5xx:     0 (  0.0%)

 REQS REQ/S    KB KB/S URL
    1  0.14  10.2  1.5*/manual/bind.html
    1  0.14   0.1  0.0 /manual/images/down.gif
    1  0.14   0.1  0.0 /manual/images/up.gif
```

Donde las cinco primeras líneas son información respecto a los procesos de APACHE:

- ✔ En la primera línea se tiene la hora del último acceso (last hit), el tiempo de ejecución del comando (atop runtime) y la hora actual.
- ✔ En la segunda línea (All) se tiene el total de solicitudes que el programa ha procesado desde su ejecución (10 reqs), el número de peticiones por segundo (0.2/sec), el total de datos transferidos (58.4K), el volumen de transferencia por segundo (1425.0B/sec) y la media de cantidad de datos por solicitud (5985.1B/req).
- ✔ En la tercera línea se tiene las medidas por código de respuesta del servidor (2xx, 3xx, 4xx y 5xx) y se tiene para cada código la cantidad de respuestas y entre paréntesis su porcentaje con respecto al total.
- ✔ En la cuarta y quinta línea, se tiene información similar a la segunda y tercera línea respectivamente, con la diferencia de que estas medidas se las realiza en un lapso de tiempo determinado por la variable R, para este ejemplo se tiene que R (30s) treinta segundos, que sería el tiempo de muestreo para la cuarta y quinta línea. Mediante los parámetros H (hits) y T (time) podremos cambiar el valor de R.
- ✔ En las líneas restantes, por debajo de la quinta línea, se tiene la información de los últimos accesos por URL, donde:
  - ✔ **REQS**: Total de solicitudes que el programa ha procesado desde su ejecución de la URL.
  - ✔ **REQ/S**: Número de peticiones por segundo.
  - ✔ **KB**: Total de datos transferidos.
  - ✔ **KB/S**: Volumen de transferencia por segundo.
  - ✔ **URL**: Es la dirección a la cual hacen referencia los 4 anteriores variables (REQS, REQ-S, KB y KB/S).

Cuando se tiene mucha información se puede utilizar filtros en esta herramienta presionando la tecla f., lo cual nos ayuda en el análisis.

Presionando la tecla d se puede cambiar la vista a URL, REFERRER ó HOST.

Para invocar a la ayuda se debe presionar la tecla f ó la tecla ?.

Conclusiones

La herramienta APACHETOP es una herramienta de monitoreo en línea (real time) el cual nos permite analizar estadísticas de acceso al servidor Web, determinando los cuellos de botella en nuestro servidor Web.

Con esta sencilla y útil herramienta APACHETOP y junto a la herramienta TOP nos permiten realizar un análisis de rendimiento, de nuestro servidor que contenga nuestra página Web.

Referencias

[1] www.webta.org/projects/apachetop/

Autor



Renzo Martinez Pardo
Ingeniero Sistemas Electrónicos
renzomp@lycos.com




Willay news
Información Actual

Willay news

LongoMatch: Digital Coach



Cuando se trata de una contienda en cualquier disciplina deportiva, los entrenadores y jugadores utilizan una serie de tácticas y estrategias, en su mayoría estudiadas con anticipación, y cuando la lucha es a ganador, se debe observar hasta el mas mínimo detalle del equipo adversario... claro que no es una exageración, de hecho muchos de los equipos que participan en los Juegos Olímpicos, Mundial de Fútbol, etc gastan un montón de dinero, preparando sus estrategias, ya sea con expertos analistas o no.

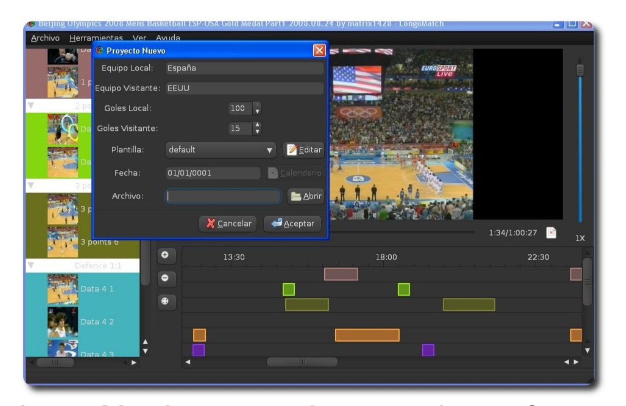


Justo en este asunto es donde entra una muy útil herramienta, se trata de LongoMatch o Digital Coach como la llama su creador Andoni Morales Alastruey, la cual resulto finalista en el 3er. Concurso Universitario de Software Libre de España.

LongoMatch es una aplicación multimedia para realizar análisis técnico/tácticos por medio de vídeos de encuentros deportivos. Ayuda a los entrenadores a identificar las jugadas clave de un partido y organizarlas por categorías: penaltis, tiros libres, defensa 1:1, salidas, oportunidades de gol, etc.

Se puede adaptar a todo tipo de deportes cuenta con reproducción en cámara lenta - frame stepping, crea listas con las jugadas mas relevantes para verlas secuencialmente, exporta una jugada tomando capturas a velocidad variable para análisis estático – archivos PNG, es multiplataforma, etc.

En el aspecto técnico, LongoMatch está escrito C# y corre bajo Mono. La interfaz gráfica usa las librerías GTK, la base de datos está implementada con db4o (una base de datos puramente orientada a objetos) y como motor gráfico usa Gstreamer.



LongoMatch con muy buenas miras al futuro, se encuentra en su versión 0.15.4, y seguramente cuando ya este lista su versión 1.0 muchos clubes deportivos incluyendo los mas pequeños, podrán tener acceso a esta tecnología, tecnología que por cierto suele ser muy cara.

GENIVI: Una Alianza Poderosa

Desde que la exitosa compañía de automóviles BMW lanzo a la plataforma GENIVI de código abierto , como un elemento estratégico para mejorar al máximo la experiencia de conducir, muchas empresas a lo largo de este año se aliaron a este gran proyecto, apoyando a la nueva era de cooperación entre fabricantes y proveedores de tecnología, cuyo interés compartido es el de agilizar el desarrollo de IVI (In Vehicle Infotainment – Info-entretenimiento en los vehículos)



IVI es un campo que esta avanzando a pasos gigantes debido a que abarca productos y servicios automotrices y de entretenimiento, incluyendo: música, noticias, Internet, multimedia, navegación y localización, además de telefonía. Por lo general si un fabricante de automóviles pretende que un determinado modelo tenga este tipo de tecnología deberá: desarrollarlo, probarlo, implementarlo y además dar soporte a estos productos... tanto en de Hardware y Software parecidos a las de una computadora.

Los fundadores de GENIVI BMW, Wind River, Intel, GM, PSA, Delphi, Magneti-Marelli, y Visteon: creen en el poder de código abierto y en el desarrollo de software de colaboración, para crear un eco-sistema que fomente la innovación, acorte los ciclos de desarrollo, reduzca el tiempo de comercialización y costes para las empresas de desarrollo de equipos y software.

GENIVI cuenta con nuevos miembros: Nokia que contribuirá con su amplia experiencia en la normalización y la creación de especificaciones abiertas, NEC Electronics traerá su experiencia en dispositivos semiconductores y ayudara en el desarrollo de plataformas abiertas, Mentor Graphics y Wipro Technologies.



Sin duda alguna estamos viviendo una era bastante diferente o quizá soñada y aunque por ahora GENIVI se centre en tecnologías solo para el entretenimiento y no así para el mantenimiento o control del vehículo, en menos de lo que nos imaginemos, esta Poderosa Alianza, nos estará impresionando con interesantes noticias sobre esta grandiosa plataforma.

Autor



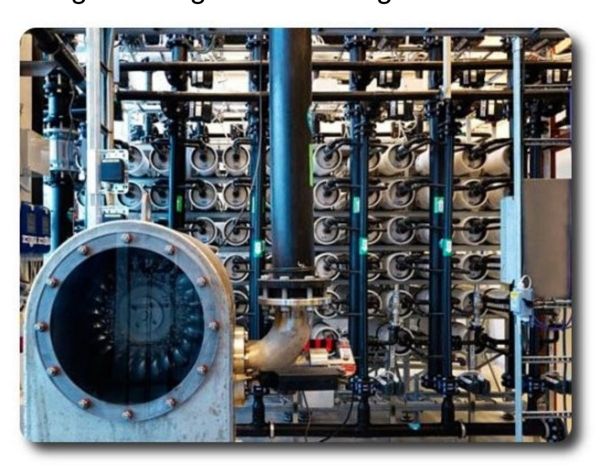
Marcia Estrella Velasquez Aguilar
Egresada de la carrera de Ingeniería de Sistemas (F.N.I.)
mevaguerreradelaluz@gmail.com

Willay news

Energía Osmótica

En números anteriores ya habíamos comentado que muchas empresas orientaban su investigación a la producción de energía alternativa y también al cuidado del medio ambiente, otros dos claros ejemplos de esto son:

En Noruega, específicamente en las plantas de Statkraft existe una planta que genera electricidad gracias a un grupo de células que fuerzan la ósmosis: “la planta genera energía explotando la que está disponible al mezclarse el agua dulce con el agua del mar. La energía osmótica es renovable y es libre de emisiones, está siendo desarrollada por Statkraft desde hace 10 años y será capaz de contribuir con la producción de energía ecológica en forma global”.



En la actualidad esta tecnología aun se sigue perfeccionando, pero según Statkraft “el potencial global de energía osmótica es estimado en 1600-1700 TWh por año” y las plantas se pueden instalar en cualquier lugar en que agua dulce y salada se mezclen.

Esta podría representar una innovadora fuente de energía alternativa en el mundo

Watersquare almacena agua de lluvia

Según expertos alemanes es lamentable ver como se van litros y litros de agua cuando llueve, por lo cual este grupo de expertos de Urbanisten ha ideado la Watersquare que es como un gran contenedor de agua de lluvia, que almacena el agua proveniente de la lluvia.



Con este emprendimiento los niños podrán disfrutar jugando alrededor de la “piscina” y en invierno será usada como pista de patinaje sobre hielo. Incluso se estudia aun el uso posterior que se le pueda dar al agua almacenada.

Diseños innovadores

Continúan los diseños innovadores sobre dispositivos o periféricos para ordenadores. Es el caso del Keystuck inventado por Yoonsang Kim y Eunsung Park.



Lo sorprendente de este teclado es la capacidad que tiene para plegarse como si fuera un abanico, para poder así trasladarlo a todos lados. Tiene conexión inalámbrica a la PC y es ideal para que ellos que cuentan con un pequeño espacio.



Otro interesante invento va referido al EZPower Universal Remote, es un mando a distancia que basa su funcionamiento en base a aire, este invento permitirá dejar la dependencia que existe en el uso de baterías por parte de los mandos a distancia, más aun cuando se dispone de una gran cantidad de estos.

La carga completa del EXPower dura unos 7 días y el dispositivo está diseñado para controlar hasta 6 aparatos (TV, VCR, Cable, SAT, AUX, DVD).

Autor



Jenny Saavedra López
Diseño y Edición Revista Atix
jenny.saavedra@atixlibre.org

Me agradaria
preparar en estos días,
un árbol de Navidad
muy especial
y colgar, en lugar de regalos,
los nombres
de todos mis amigos. Los de cerca
y los de más lejos. Los de siempre y los que
tengo ahora.
Los que veo cada dia, y los que encuentro de
vez en cuando.
Aquellos a los que siempre recuerdo y a los que a menudo olvido.
A los constantes y a los inconstantes. A los de las horas
alegres y a los de las horas difíciles. A los que sin querer herí,
y a los que sin querer me hirieron. Aquellos a quienes conozco
profundamente, y aquellos a quienes solo conozco por su
apariencia.
A los que me deben algo y a los que les debo mucho. A los amigos humildes
y a los amigos importantes. Por eso los nombro a todos, a todos los amigos que han
pasado por mi vida. A los que recibis este mensaje y a los que no lo recibirán.
Un arbol de raices profundas, para que vuestros nombres no se puedan arrancar jamás.
Un árbol que, al florecer el año que viene, nos traiga ilusión, salud, amor y paz.
Ojalá que por Navidad, nos podamos reencontrar compartiendo los mejores deseos
de esperanza,
dando algo
de felicidad a aquellos
que lo han perdido todo.

Feliz
Navidad



Comics

Curiosidades















Libres para pensar, libres para decidir, libres para crear



Te ofrecemos este espacio para mostrar tu Creatividad

Envíanos tus diseños y creaciones para publicarlos



Aportando al conocimiento
de la comunidad

Consultas

Contacto

Para solicitar cualquier información, puedes contactar a:

- Esteban Saavedra López (esteban.saavedra@atixlibre.org)
- Jenny Saavedra (jenny.saavedra@atixlibre.org)

Publicación

Te invitamos a ser parte de la **Revista ATIX**. La forma de participar puede ser enviándonos:

- Artículos referidos a áreas como:
  - Instalación y personalización de Aplicaciones
  - Scripting
  - Diseño gráfico
  - Programación y desarrollo de aplicaciones
  - Administración de servidores
  - Seguridad
  - y cualquier tema enmarcado dentro del uso de Software Libre
- Trucos y recetas.
- Noticias.
- Comics.
- Links de interés.



Empezamos a registrar nuestra historia



Hacia un Futuro Innovador

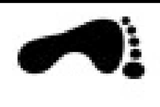


http://www.atixlibre.org

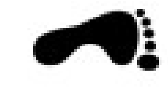
Por un Mundo Ético, Libre y Justo



Marcamos Huella








http://revista.atixlibre.org